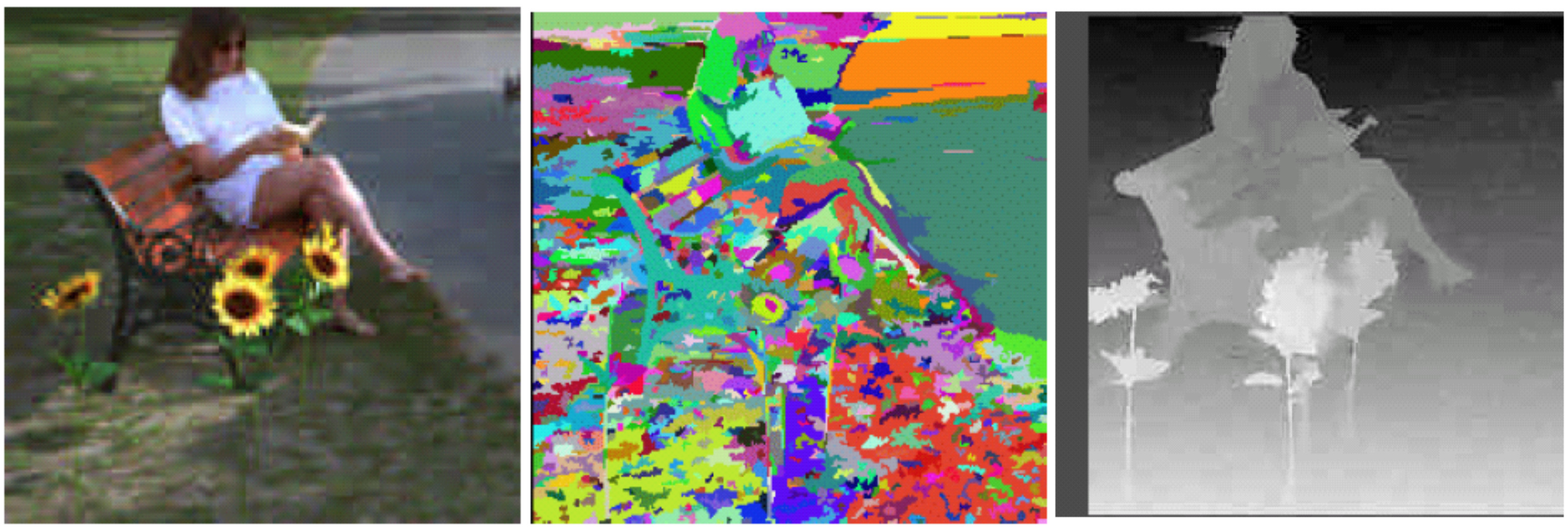
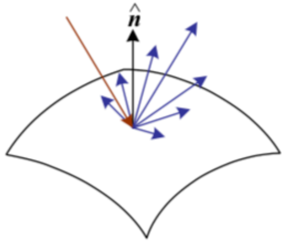
Dense Stereo



Some Slides by Forsyth & Ponce, Jim Rehg, Sing Bing Kang

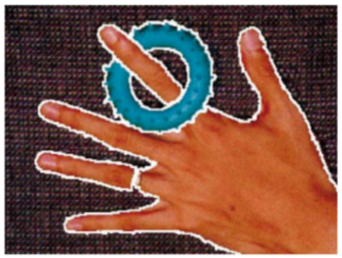
(Does not line up well with Szeliski book)



2. Image Formation

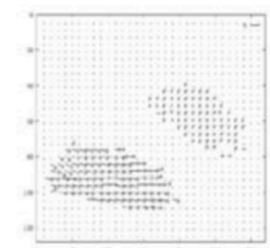
3. Image Processing

4. Features

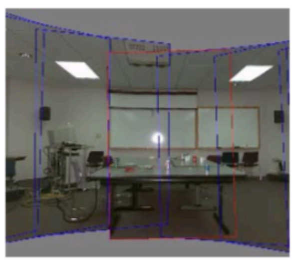


5. Segmentation

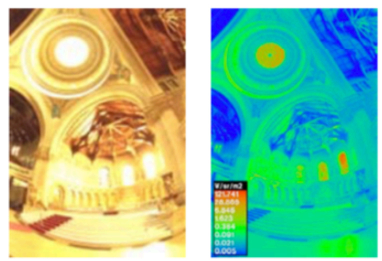
6-7. Structure from Motion



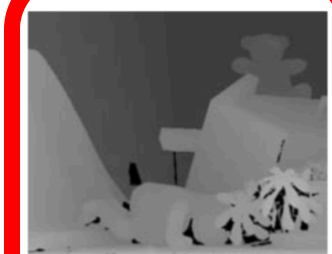
8. Motion



9. Stitching

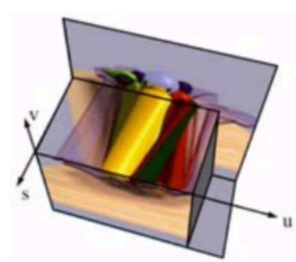


10. Computational Photography



11. Stereo

12. 3D Shape



13. Image-based Rendering

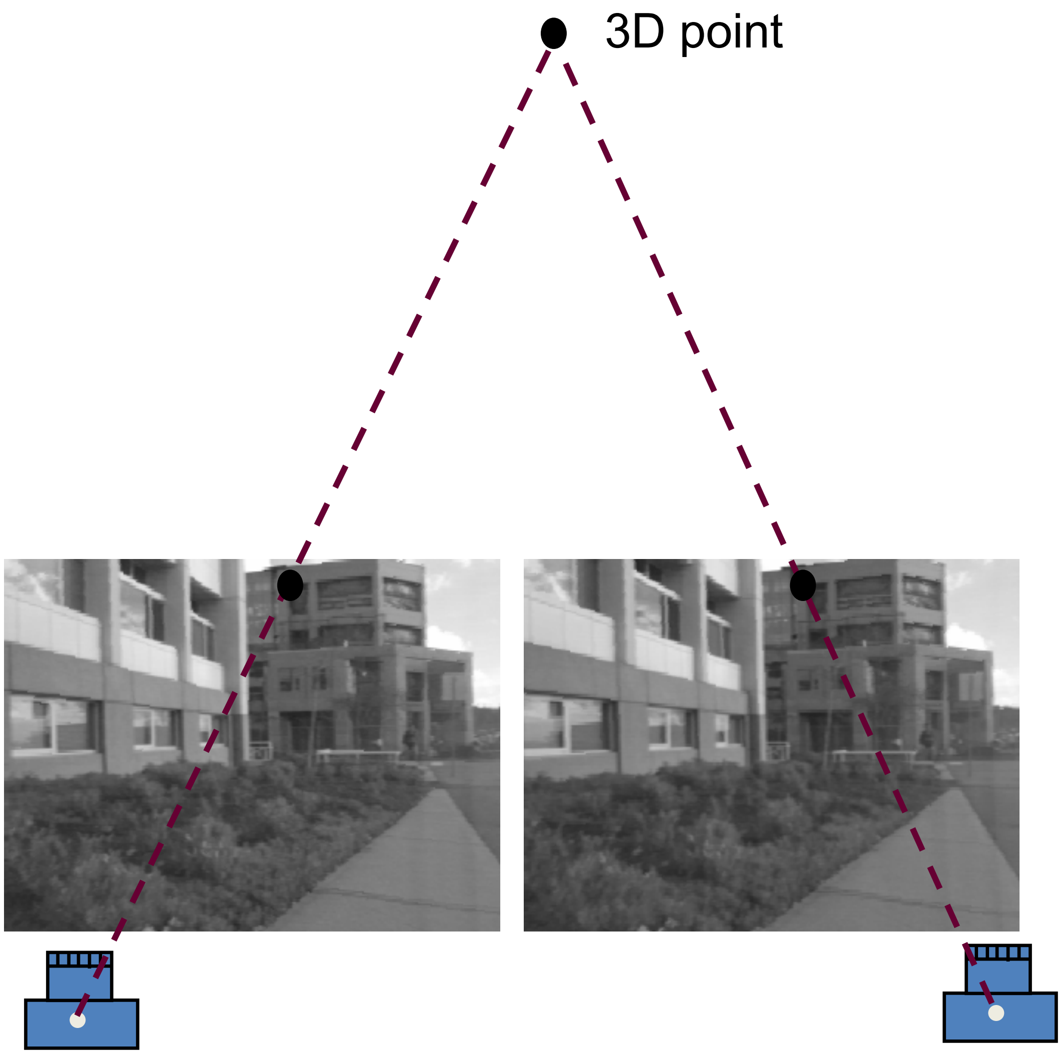
14. Recognition

Etymology

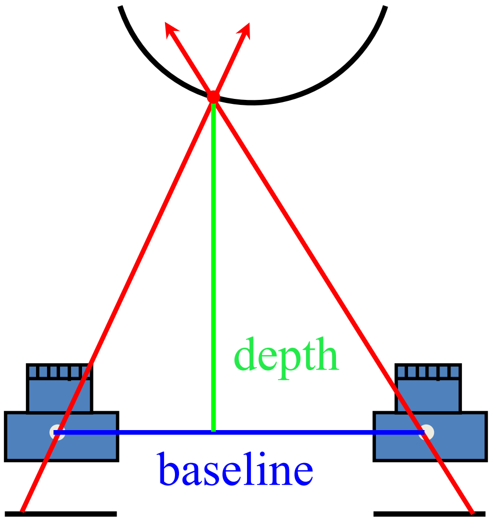
Stereo comes from the Greek word for *solid* (στερεο), and the term can be applied to any system using more than one channel

Effect of Moving Camera

- As camera is shifted (viewpoint changed):
  - 3D points are projected to different 2D locations
  - Amount of shift in projected 2D location depends on depth
- 2D shifts=Parallax



Basic Idea of Stereo



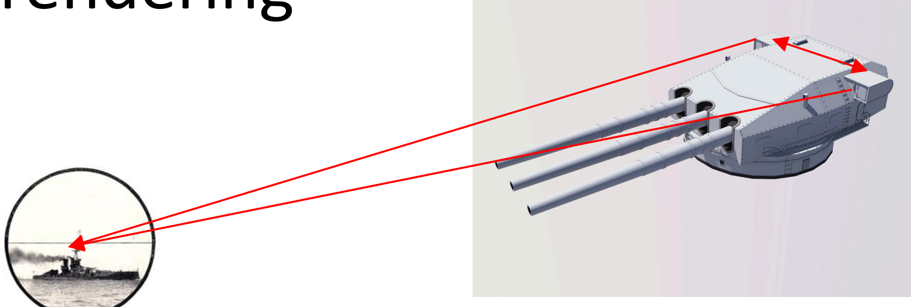
Triangulate on two images of the same point to recover depth.

- Feature matching across views
- Calibrated cameras

Matching correlation windows across scan lines

Why is Stereo Useful?

- Passive and non-invasive
- Robot navigation (path planning, obstacle detection)
- 3D modeling (shap analysis, reverse engineering, visualization)
- Photorealistic rendering

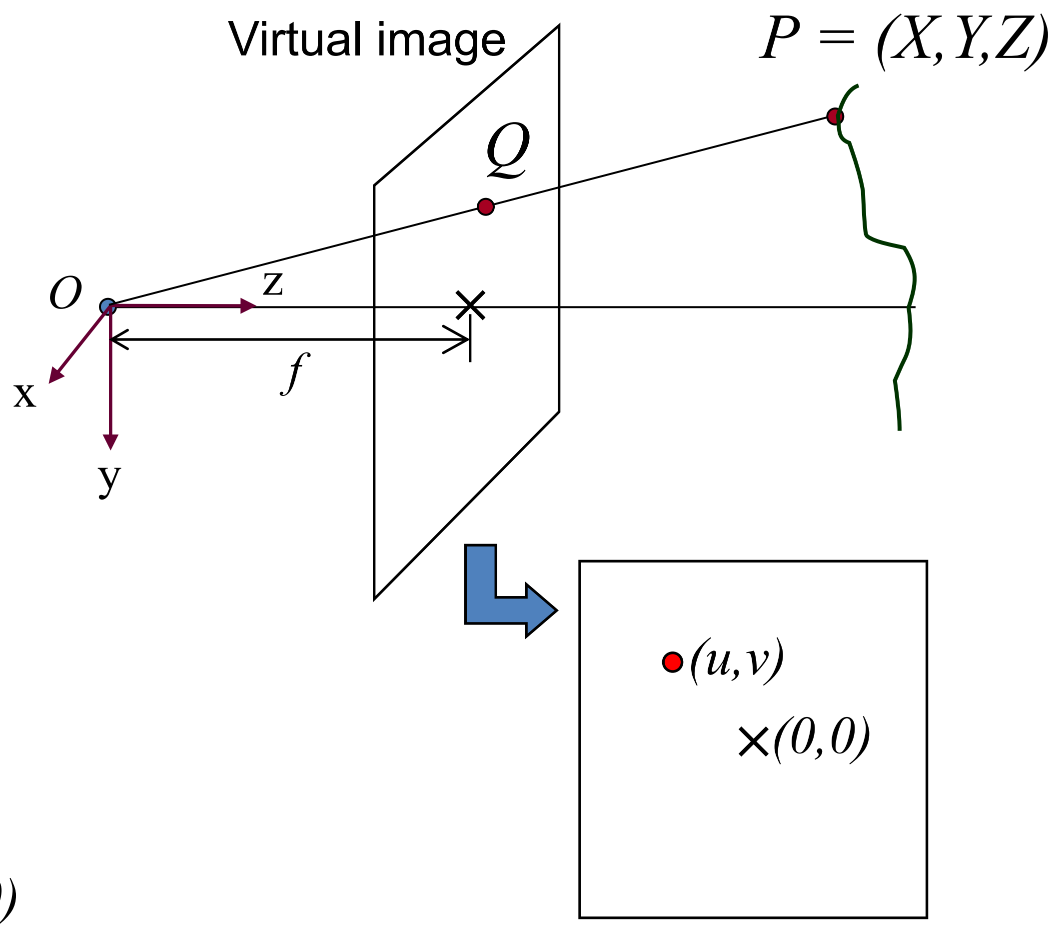


Outline

- Pinhole camera model
- Basic (2-view) stereo algorithm
  - Equations
  - Window-based matching (SSD)
  - Dynamic programming
- Multiple view stereo

Review: Pinhole Camera Model

3D scene point P is projected to a 2D point Q in the virtual image plane

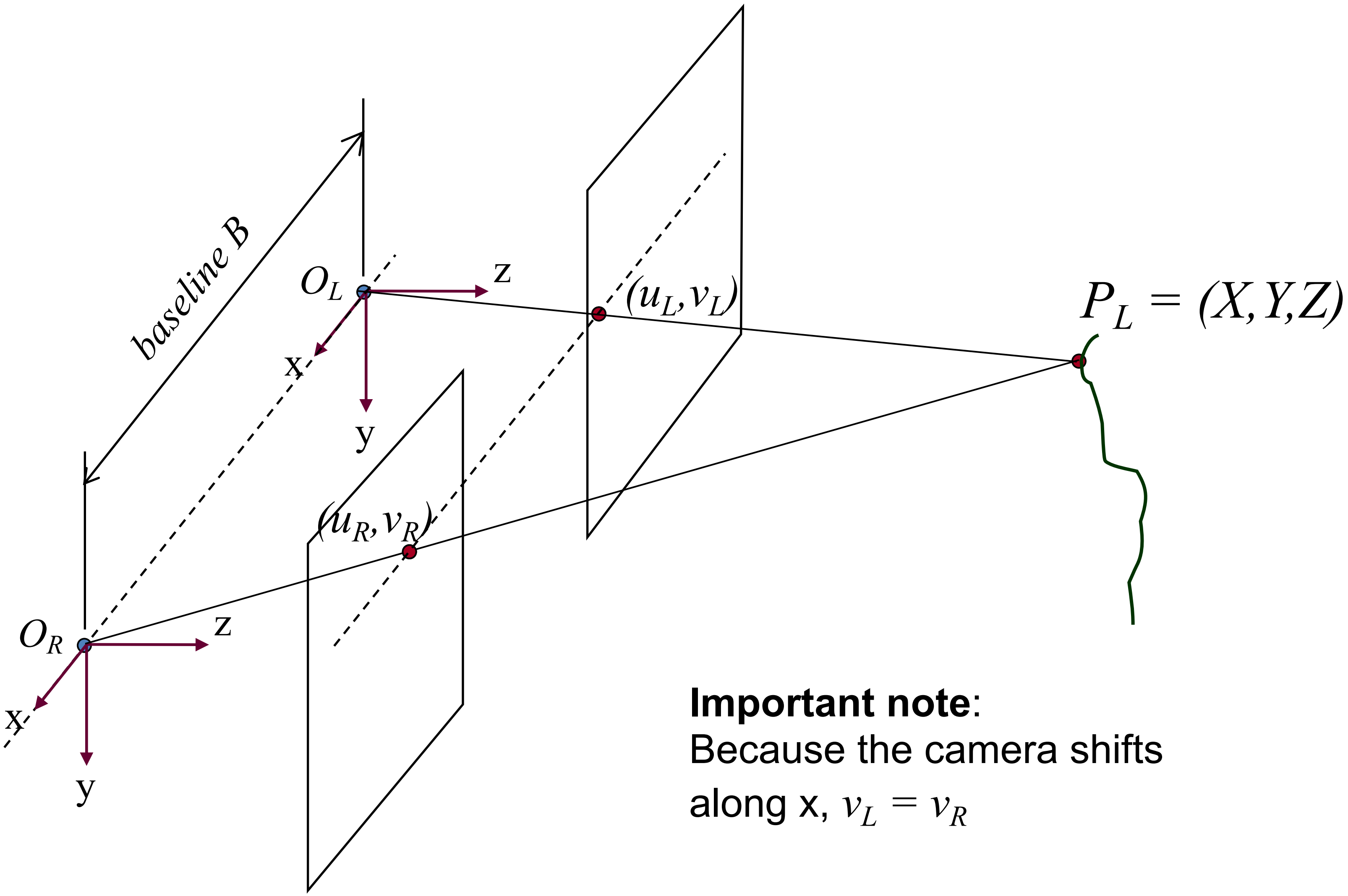


The 2D coordinates in the image are given by

$$(u, v) = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right)$$

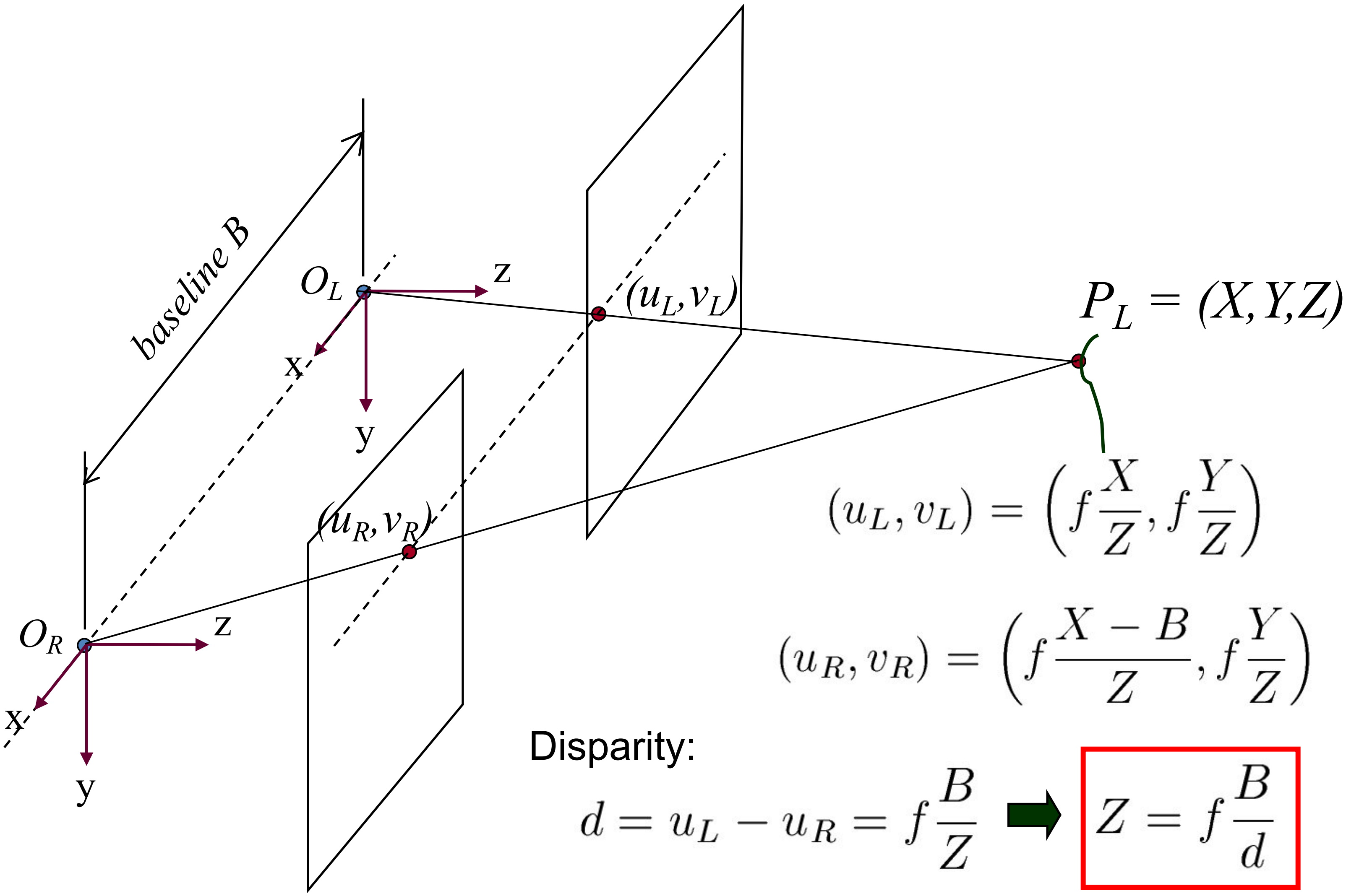
Note: image center is *(0,0)*

Basic Stereo Derivations



Important note:
Because the camera shifts along x, $v_L = v_R$

Basic Stereo Derivations



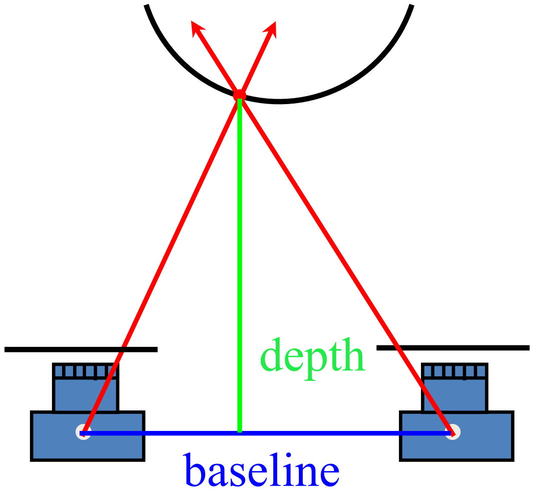
$$(u_L, v_L) = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right)$$

$$(u_R, v_R) = \left(f\frac{X-B}{Z}, f\frac{Y}{Z}\right)$$

Disparity:

$$d = u_L - u_R = f\frac{B}{Z} \Rightarrow \boxed{Z = f\frac{B}{d}}$$

Stereo Vision



$$Z(x,y) = \frac{f\,B}{d(x,y)}$$

$Z(x, y)$ is depth at pixel (x, y)
$d(x, y)$ is disparity

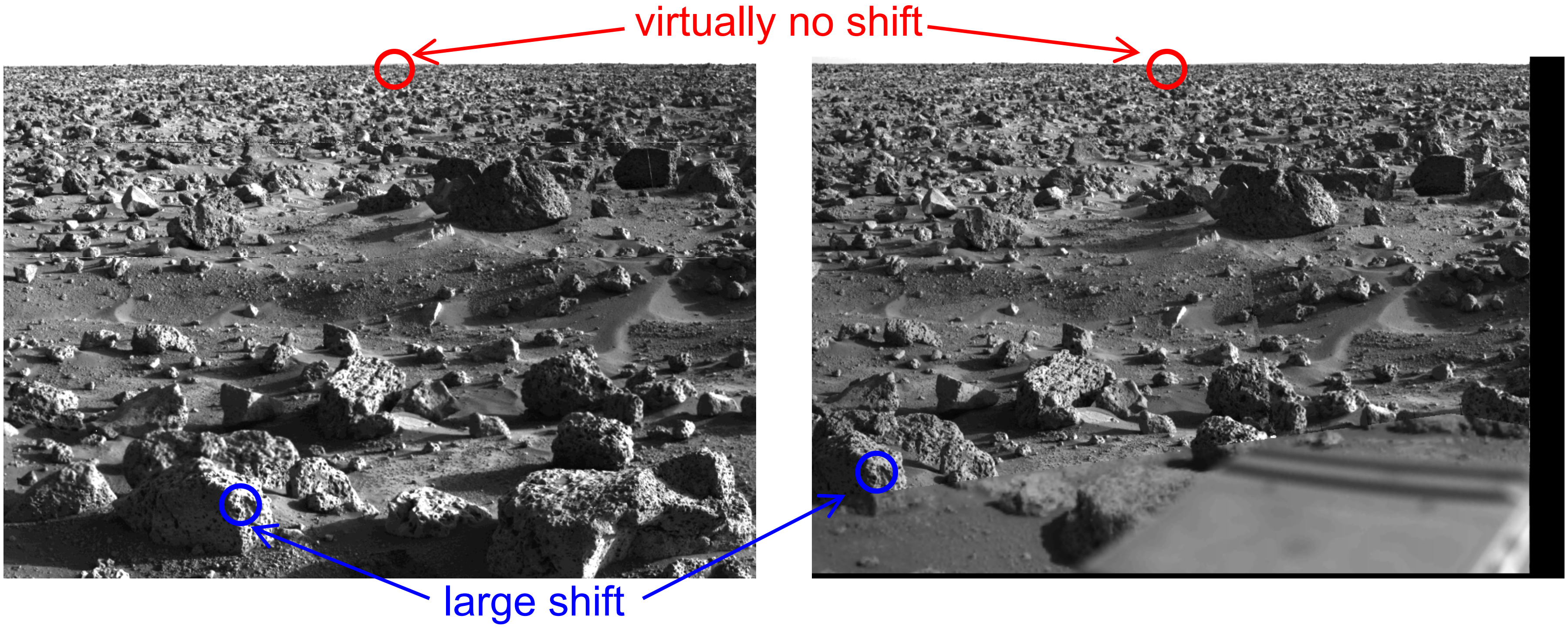
Matching correlation windows across scan lines

Components of Stereo

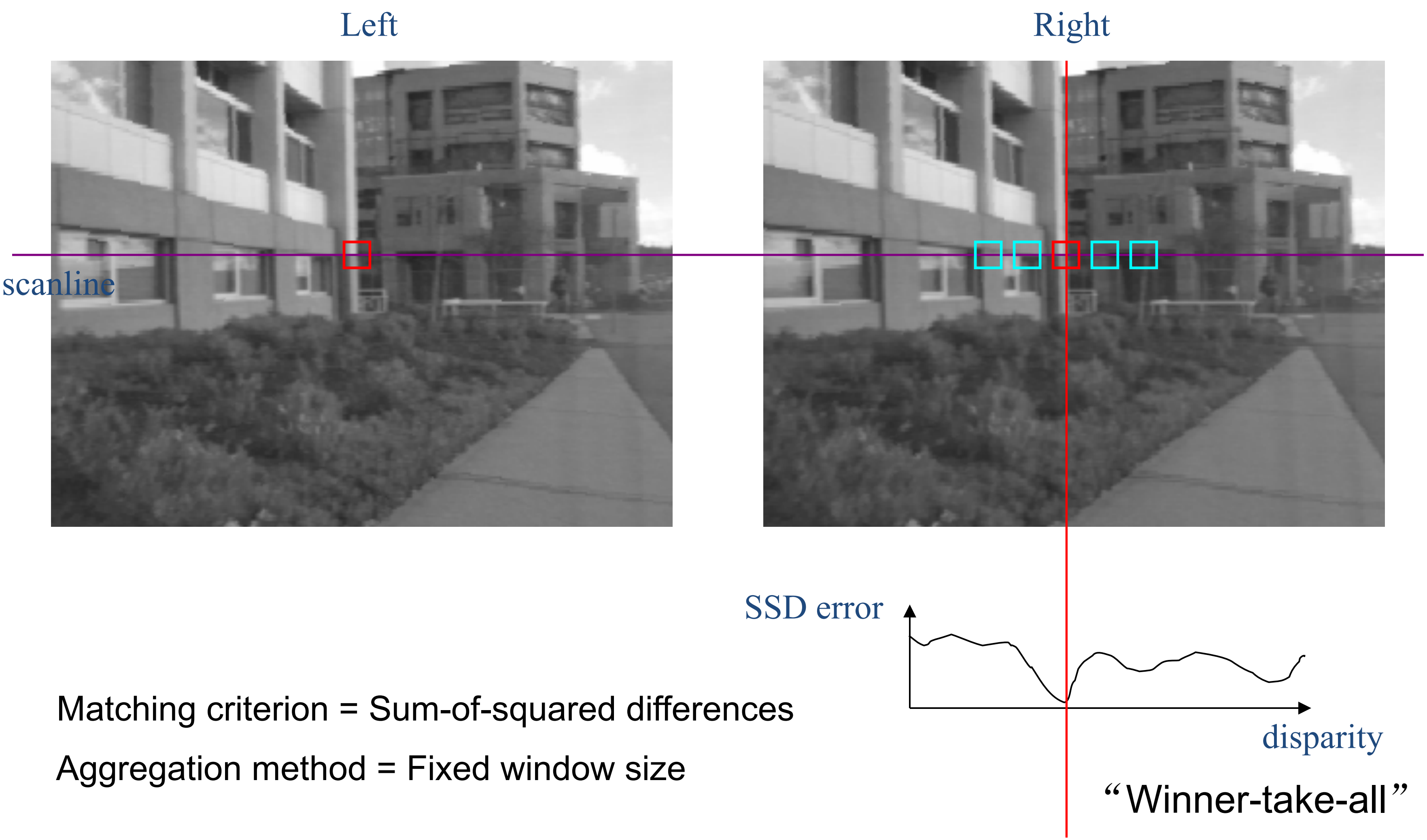
- Matching criterion (error function)
  - Quantify similarity of pixels
  - Most common: direct intensity difference
- Aggregation method
  - How error function is accumulated
  - Options: Pixel, edge, window, or segmented regions
- Optimization and winner selection
  - Examples: Winner-take-all, dynamic programming, graph cuts, belief propagation

Stereo Correspondence

- Search over disparity to find correspondences
- Range of disparities can be large



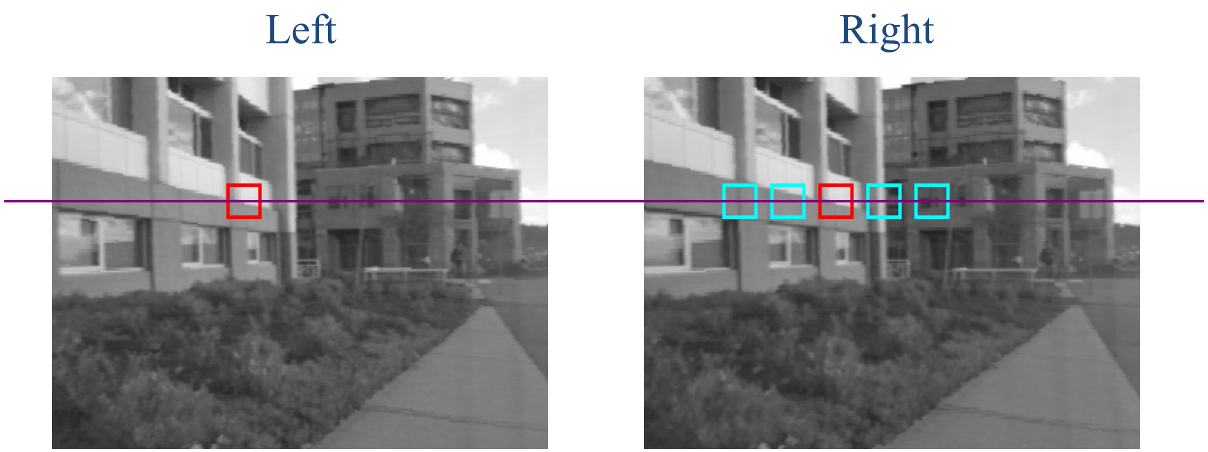
Correspondence Using Window-based Correlation



Matching criterion = Sum-of-squared differences

Aggregation method = Fixed window size

Sum of Squared (Intensity) Differences



w_L and w_R are corresponding m by m windows of pixels.

We define the window function:

$$W_m(x,y) = \{u,v \mid x - \tfrac{m}{2} \leq u \leq x + \tfrac{m}{2}, y - \tfrac{m}{2} \leq v \leq y + \tfrac{m}{2}\}$$

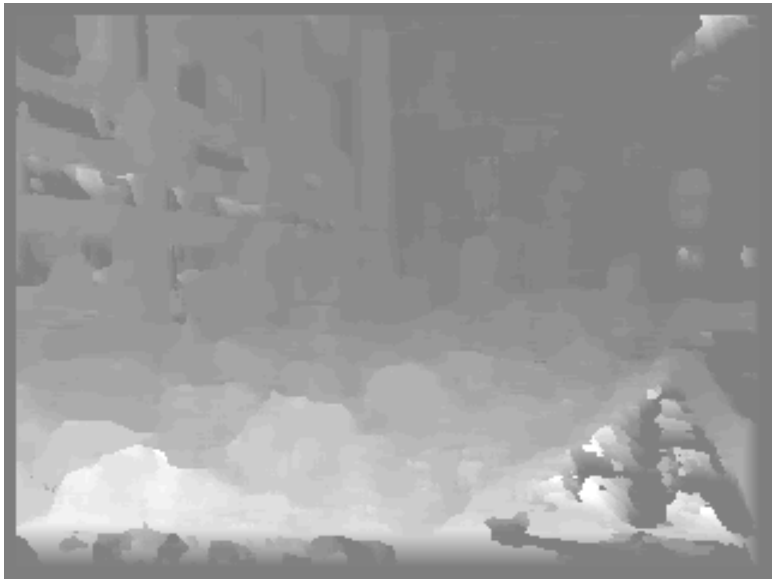
The SSD cost measures the intensity difference as a function of disparity:

$$C_r(x,y,d) = \sum_{(u,v) \in W_m(x,y)} [I_L(u,v) - I_R(u-d,v)]^2$$

Correspondence Using Correlation

Left

Disparity Map



Images courtesy of Point Grey Research

Image Normalization

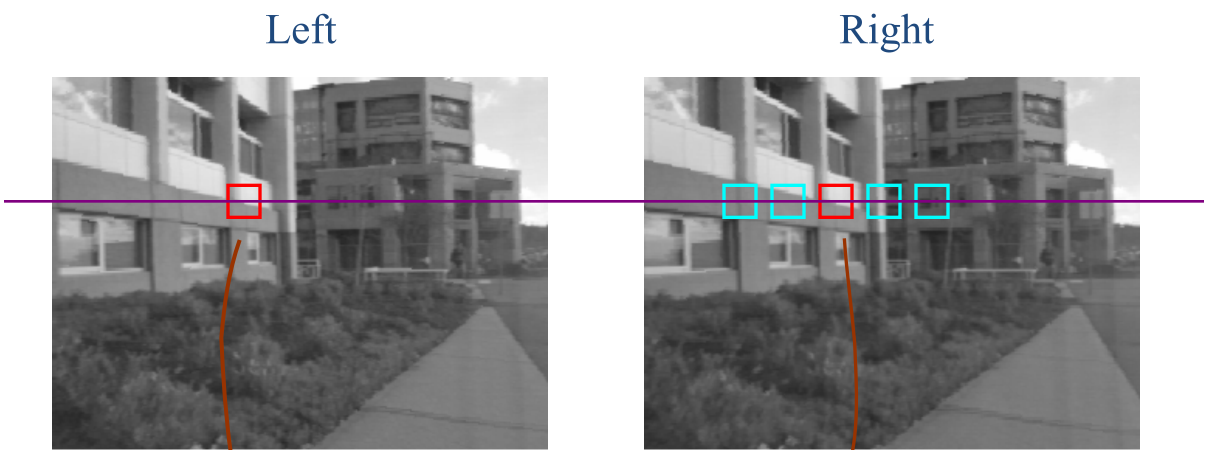
- Images may be captured under different exposures (gain and aperture)
- Cameras may have different radiometric characteristics
- Surfaces may not be Lambertian
- Hence, it is reasonable to normalize pixel intensity in each window (to remove bias and scale):

$$\bar{I} = \frac{1}{|W_m(x,y)|} \sum_{(u,v)\in W_m(x,y)} I(u,v)$$ Average pixel

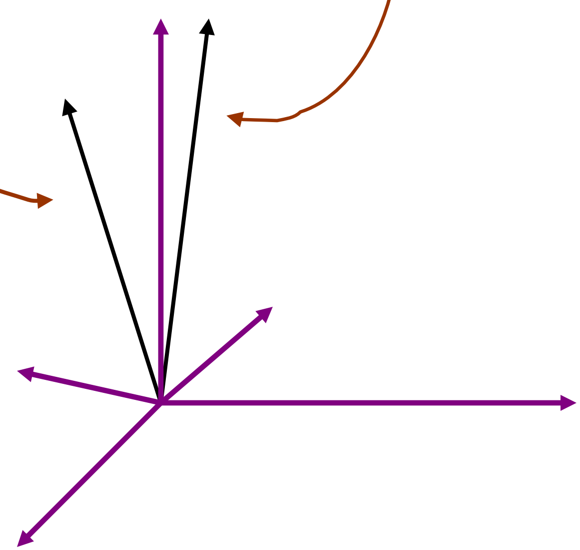
$$\|I\|_{W_m(x,y)} = \sqrt{\sum_{(u,v)\in W_m(x,y)} [I(u,v)]^2}$$ Window magnitude

$$\hat{I}(x,y) = \frac{I(x,y) - \bar{I}}{\|I - \bar{I}\|_{W_m(x,y)}}$$ Normalized pixel

Images as Vectors



"Unwrap" image to form vector, using raster scan order



Each window is a vector in an m^2 dimensional vector space. Normalization makes them unit length.

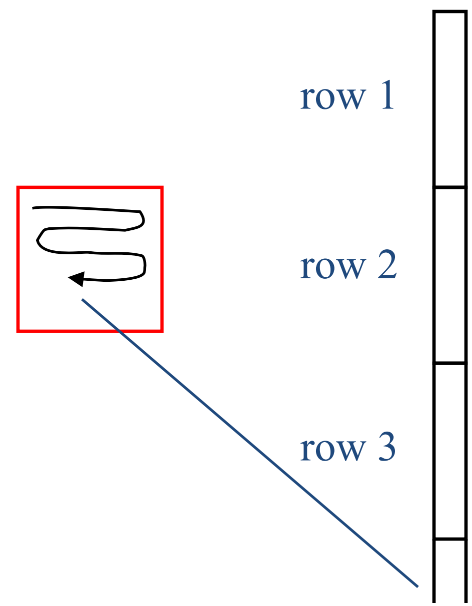
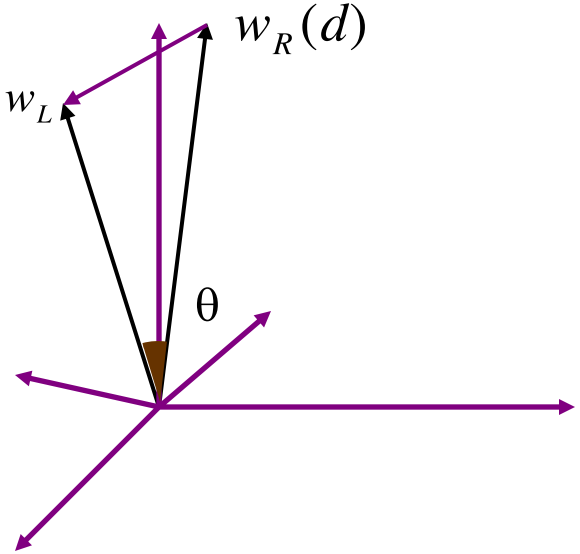


Image Metrics



(Normalized) Sum of Squared Differences

$$C_{\mathrm{SSD}}(d) = \sum_{(u,v)\in W_m(x,y)} [\hat{I}_L(u,v) - \hat{I}_R(u-d,v)]^2$$

$$= \left\| w_L - w_R(d) \right\|^2$$

Normalized Correlation

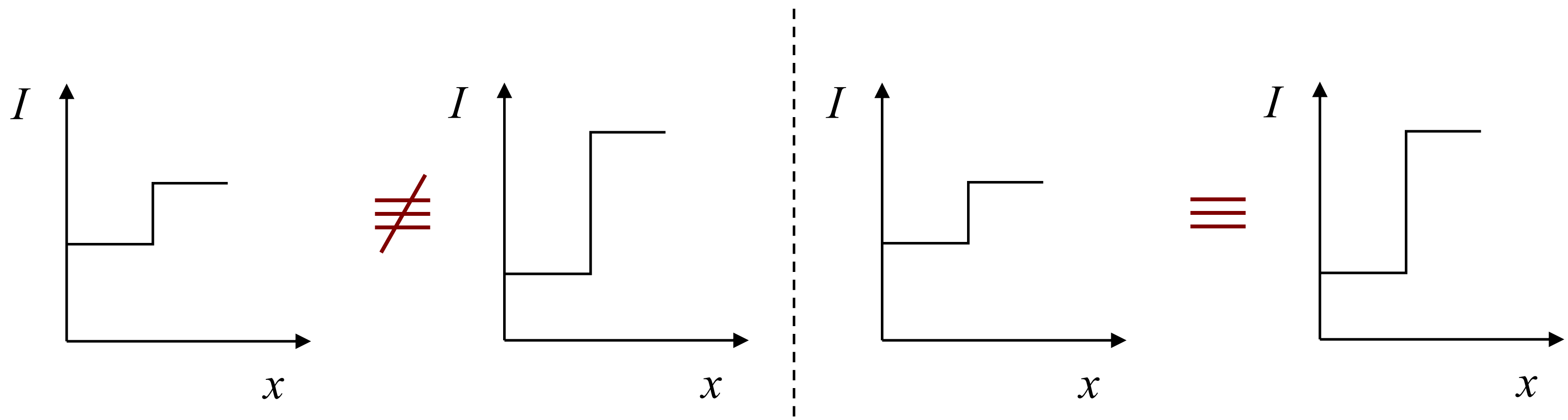
$$C_{\mathrm{NC}}(d) = \sum_{(u,v)\in W_m(x,y)} \hat{I}_L(u,v)\hat{I}_R(u-d,v)$$

$$= w_L \cdot w_R(d) = \cos\theta$$

$$d^* = \arg\min_d \left\| w_L - w_R(d) \right\|^2 = \arg\max_d w_L \cdot w_R(d)$$

Caveat

- Image normalization should be used *only* when deemed necessary
- The equivalence classes of things that look "similar" are substantially larger, leading to more matching ambiguities

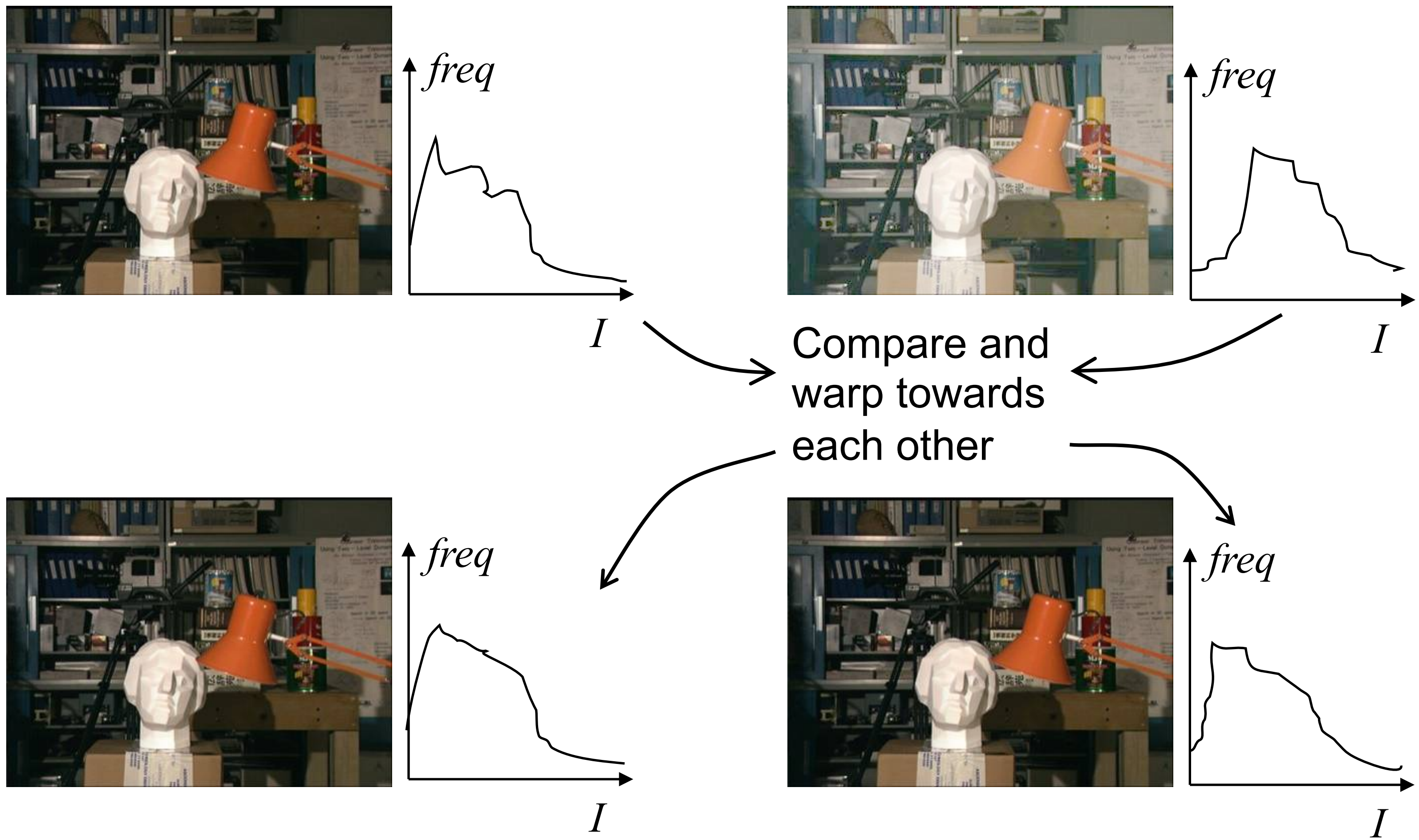


Direct intensity

Normalized intensity

Alternative: Histogram Warping

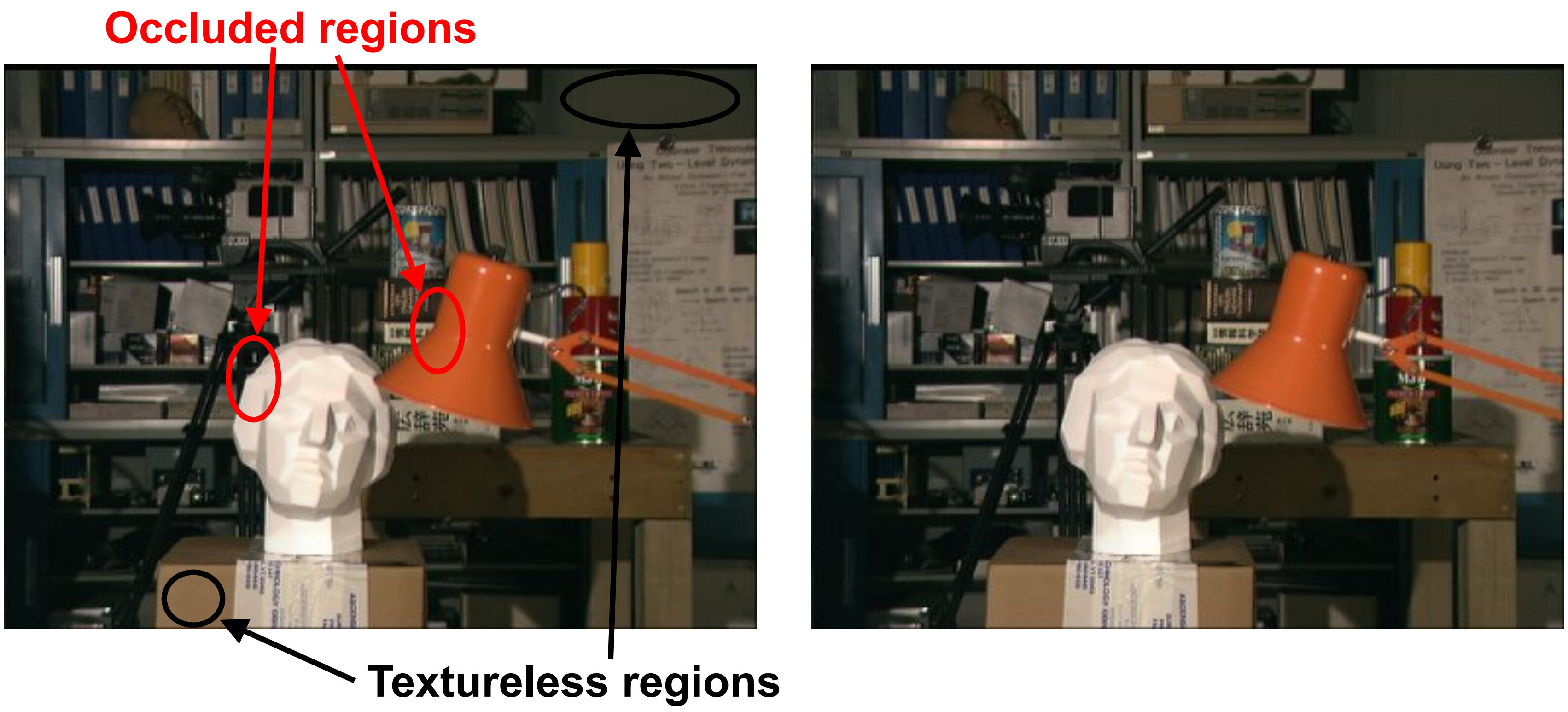
(Assumes significant visual overlap between images)



Cox, Roy, & Hingorani’ 95: “Dynamic Histogram Warping”

Two major roadblocks

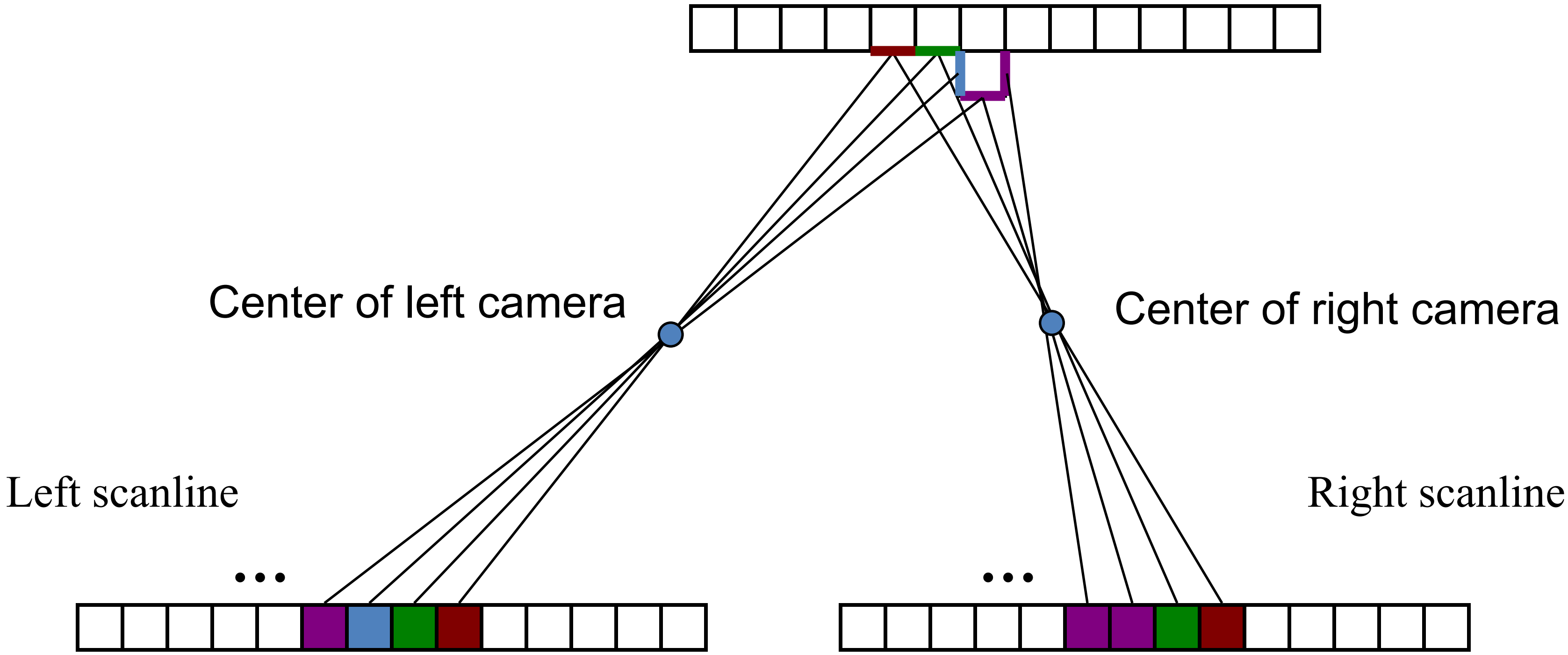
- Textureless regions create ambiguities
- Occlusions result in missing data



Dealing with ambiguities and occlusion

- Ordering constraint:
  - Impose same matching order along scanlines
- Uniqueness constraint:
  - Each pixel in one image maps to unique pixel in other
- Can encode these constraints easily in dynamic programming

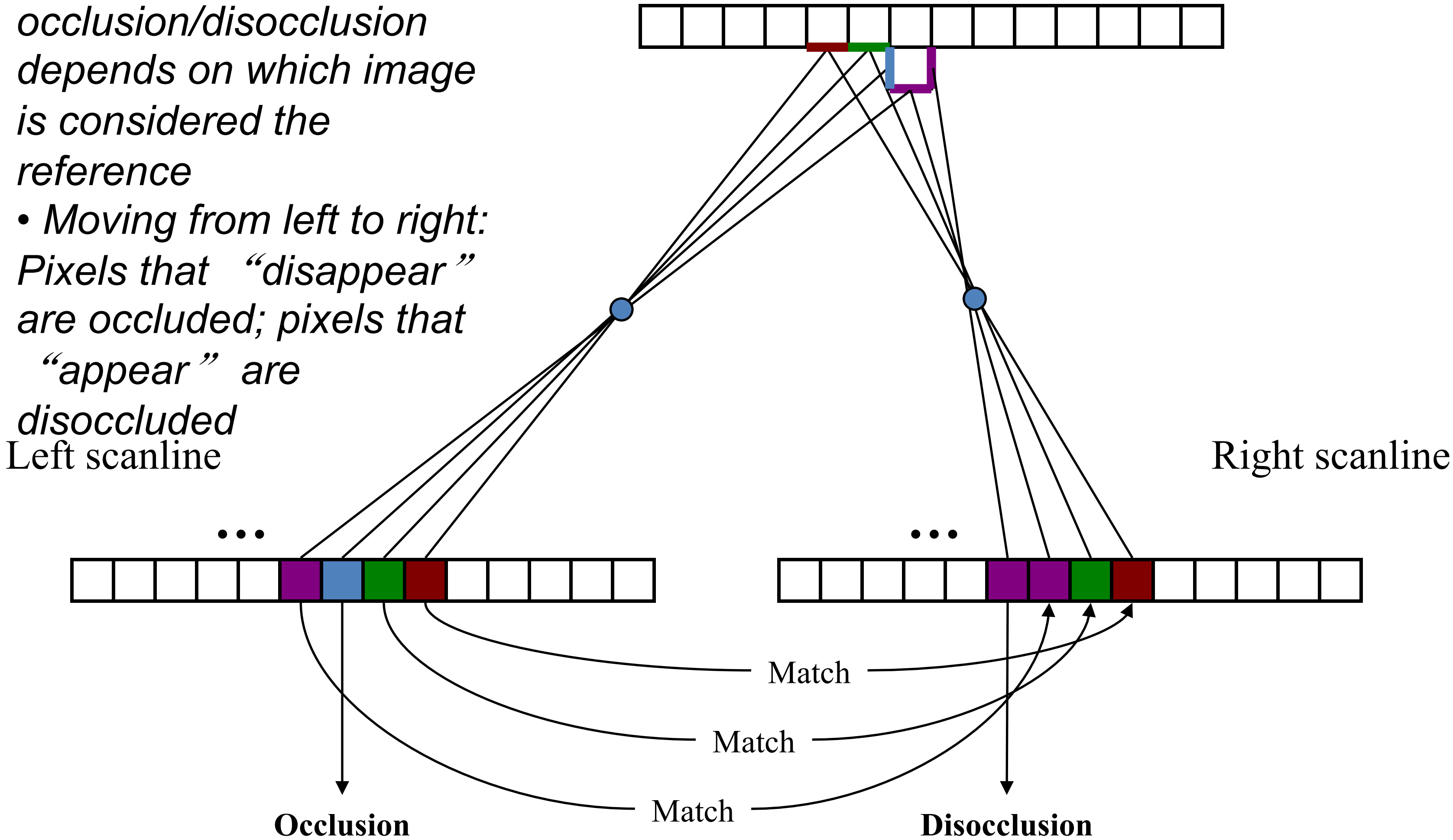
Pixel-based Stereo



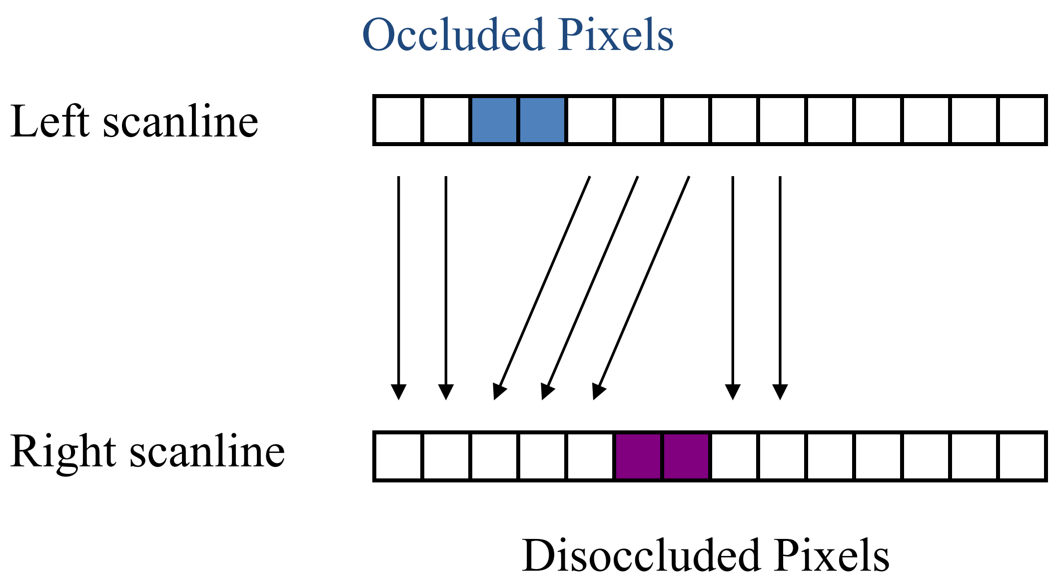
(NOTE: I’m using the actual, not virtual, image here.)

Stereo Correspondences

- *Right image is reference*
- *Definition of occlusion/disocclusion depends on which image is considered the reference*
- *Moving from left to right: Pixels that "disappear" are occluded; pixels that "appear" are disoccluded*



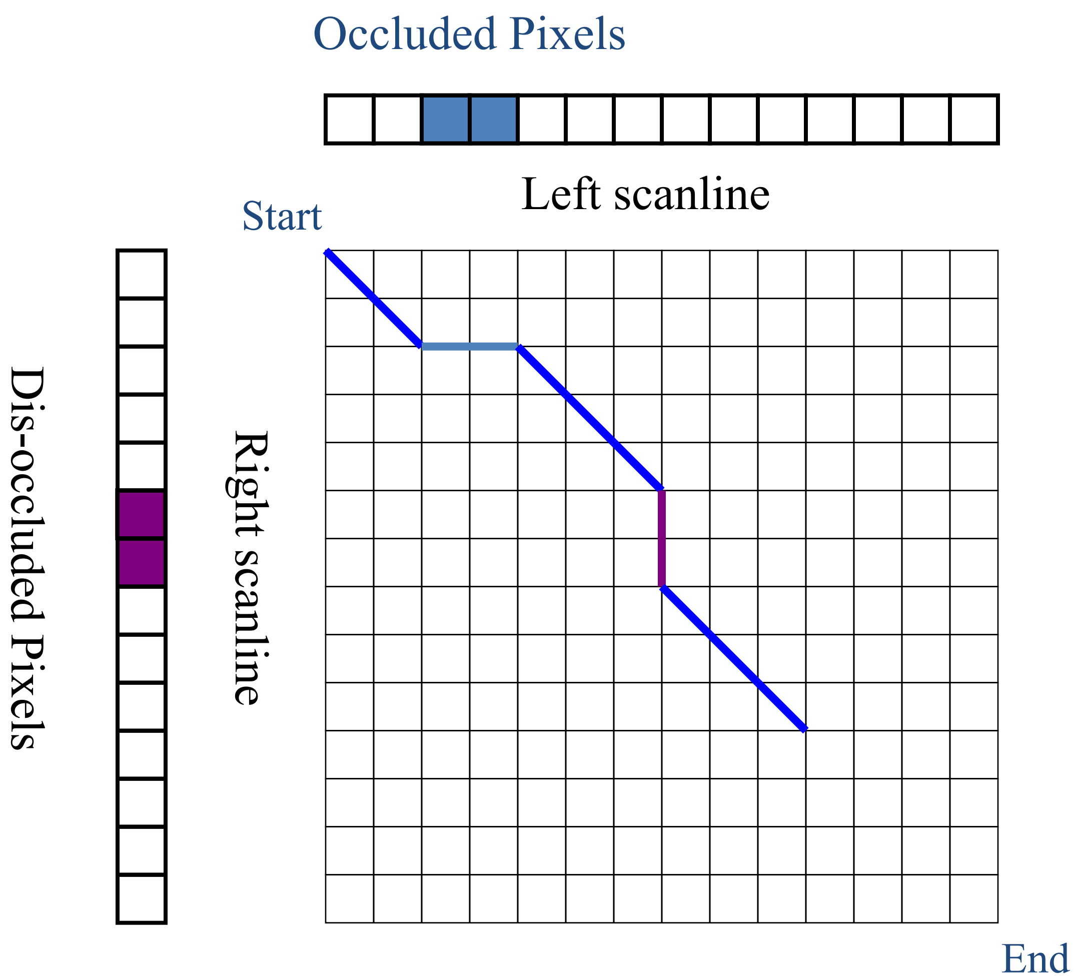
Search Over Correspondences



Three cases:

- Sequential – cost of match
- Occluded – cost of no match
- Disoccluded – cost of no match

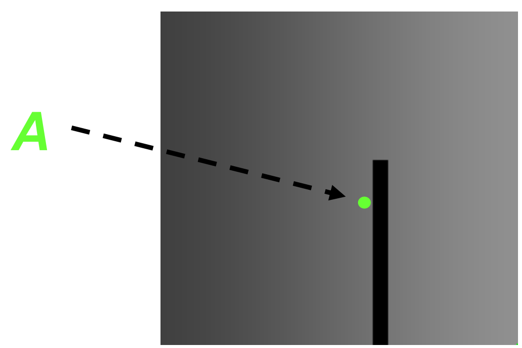
Stereo Matching with Dynamic Programming



Dynamic programming yields the optimal path through grid. This is the best set of matches that satisfy the ordering constraint

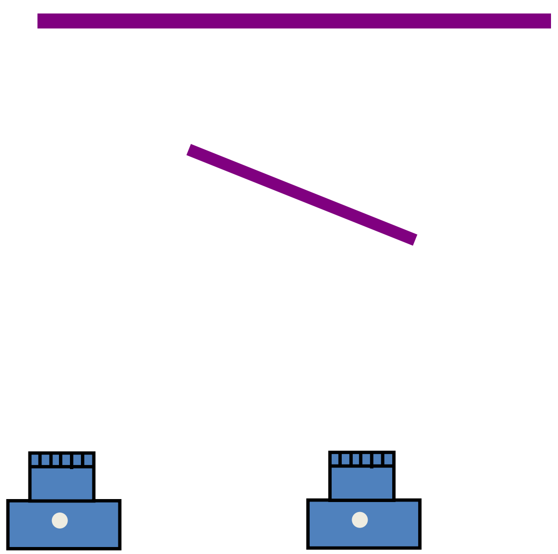
Ordering Constraint is not Generally Correct

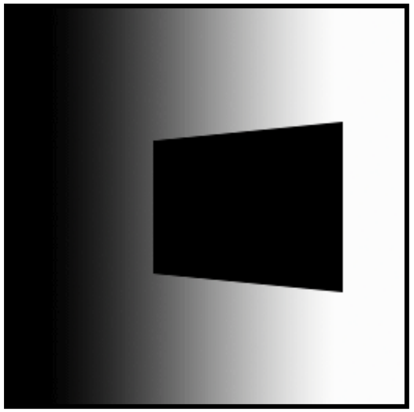
- Preserves matching order along scanlines, but cannot handle “double nail illusion”



Uniqueness Constraint is not Generally Correct

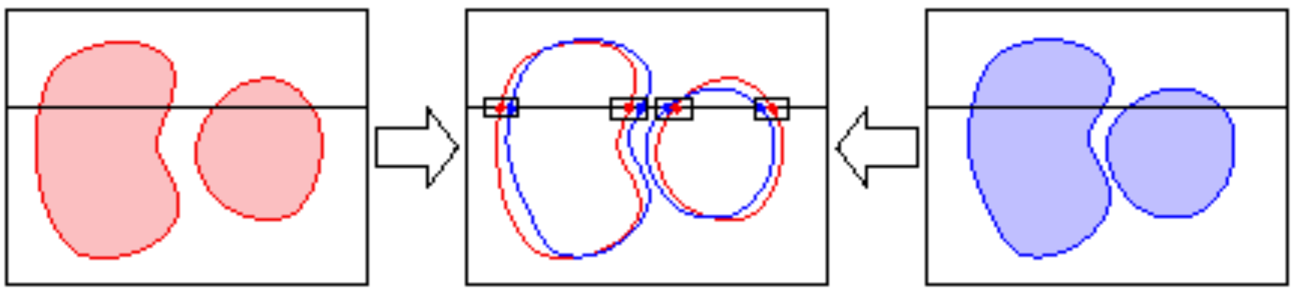
- Slanted plane: Matching between M pixels and N pixels





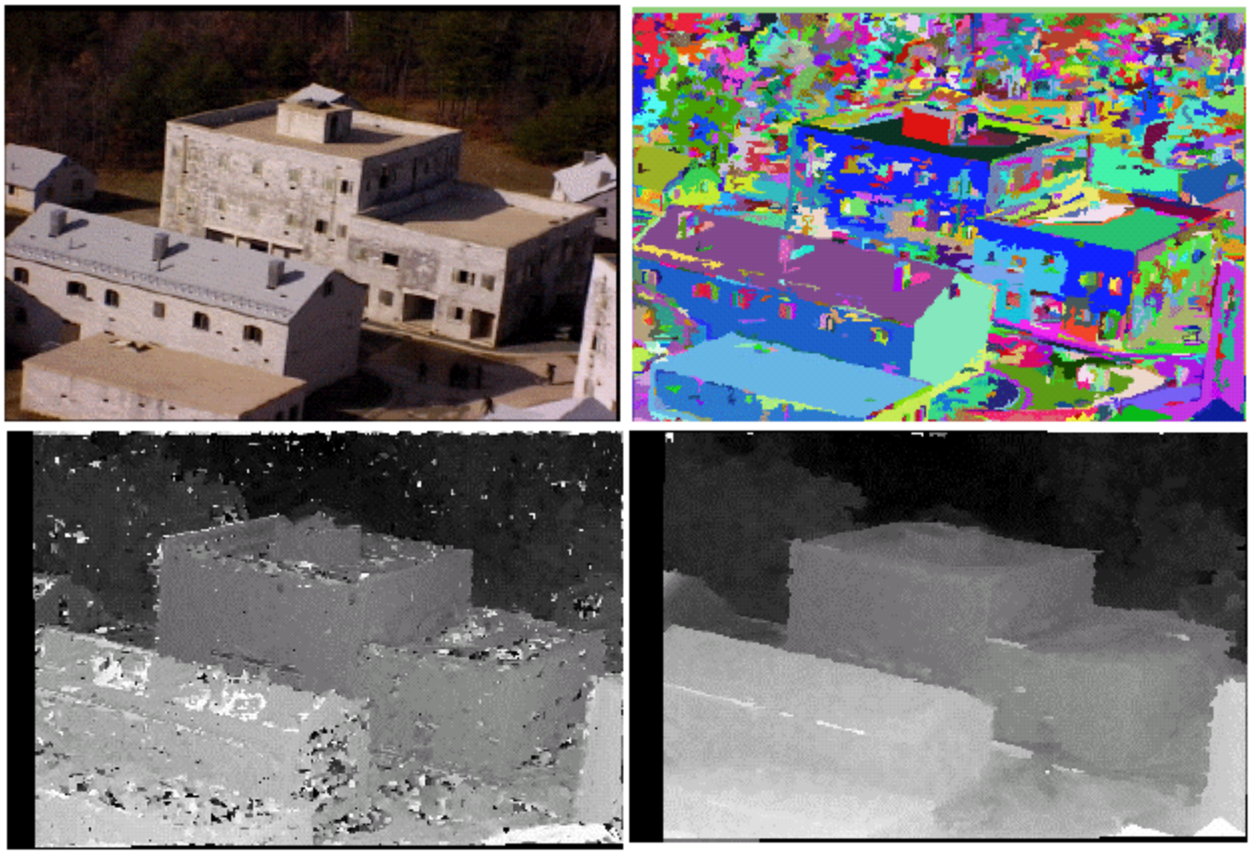
Edge-based Stereo

- Another approach is to match *edges* rather than windows of pixels:



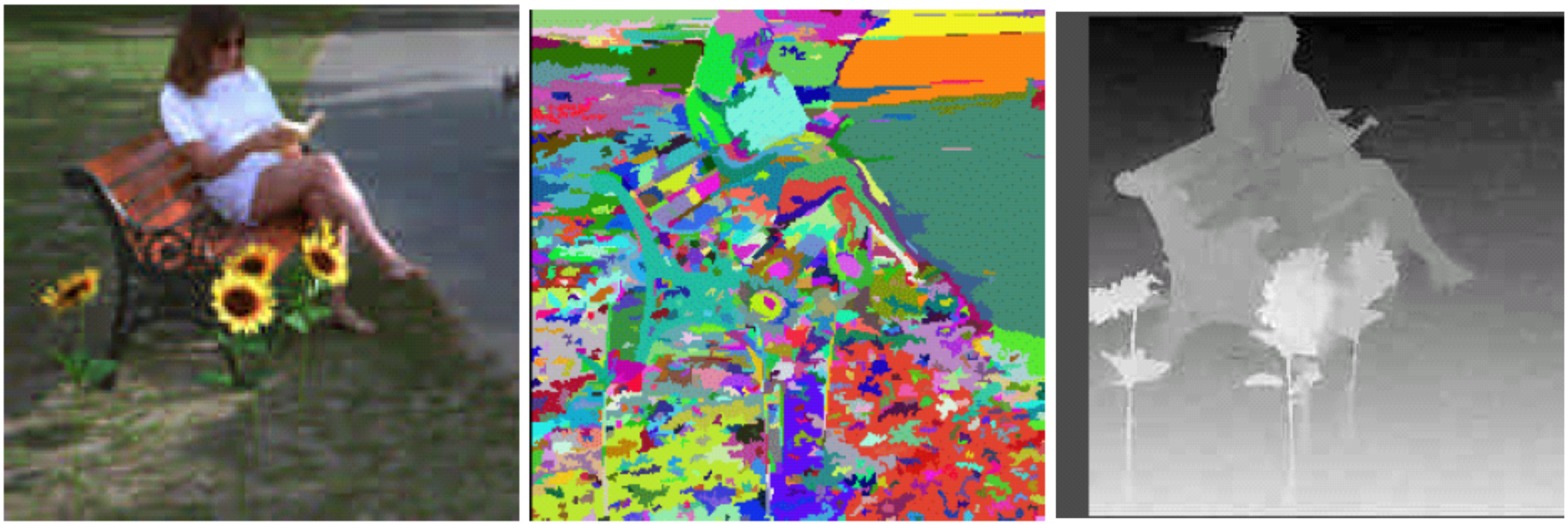
- Which method is better?
  - Edges tend to fail in dense texture (outdoors)
  - Correlation tends to fail in smooth featureless areas
  - Sparse correspondences

Segmentation-based Stereo



Hai Tao and Harpreet W. Sawhney

Another Example

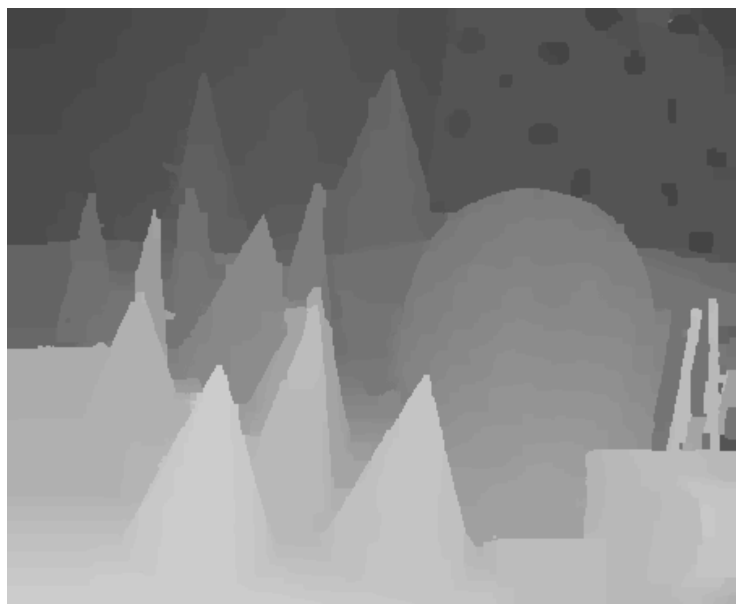


Hallmarks of
A Good Stereo Technique

- Should not rely on order and uniqueness constraints
- Should account for occlusions
- Should account for depth discontinuity
- Should have reasonable shape priors to handle textureless regions (e.g., planar or smooth surfaces)
- Should account for non-Lambertian surfaces
- There's a database with ground truth for testing: http://cat.middlebury.edu/stereo/data.html

Left

Right

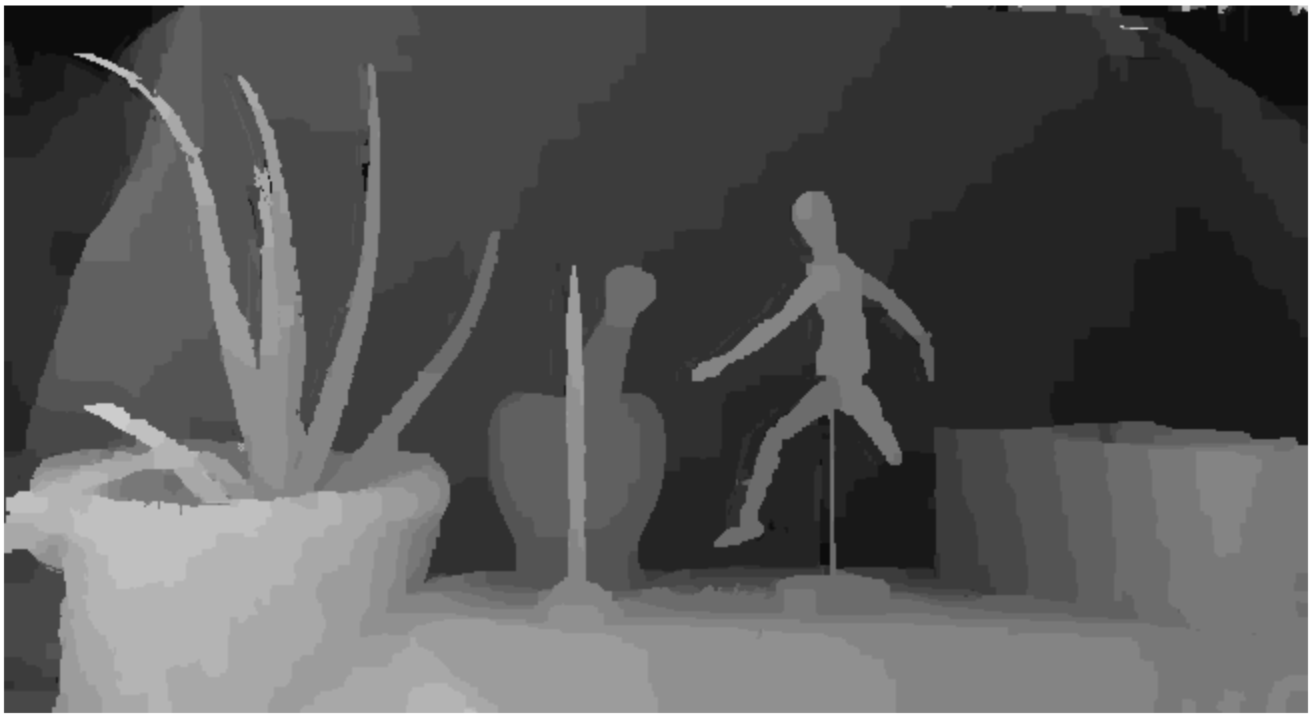


Disparity Map

Result of using a more sophisticated stereo algorithm

View Interpolation

Result using a good technique



Right image
Left image
Disparity

View Interpolation

Bottom Line: Stereo is Still Unresolved

- Depth discontinuities
- Lack of texture (depth ambiguity)
- Non-rigid effects (highlights, reflection, translucency)

From 2 views to >2 views

- More pixels voting for the right depth
- Statistically more robust
- However, occlusion reasoning is more complicated, since we have to account for *partial occlusion*:
  - Which subset of cameras sees the same 3D point?